

Real Time Systems
Professor Rajib Mall
Department of Computer Science and Engineering
Indian Institute of Technology Kharagpur
Lecture 12
Frame Size Constraints

Welcome to this lecture, we have been discussing for last one or two lectures about clock driven scheduling. Clock driven scheduling is used in simple small embedded systems, but then their number is very high and we discussed three clock driven schedulers, one is the round robin, which is not really a real time scheduler, because it does not do anything special to meet the deadlines of the real time tasks.

We looked at the table driven scheduler, the basic table driven scheduler, extremely simple concept, the schedule table and then we store the schedule table for a certain duration which is given by LCM of the periods of the tasks, periodic tasks and each time, a one shot timer is set, each time the scheduler is invoked, based on a timer alarm, it selects the next task and sets the one shot timer.


And this is executive, it is not an operating system, because it does not have many features only for simple applications, very simple scheduler which itself is the operating system. But then the main problem here is that it has a lot of overhead in terms of setting the one shot timer, each time there is the alarm timer, alarm it sets the time set again, so we had improvised scheduler which we called cyclic scheduler.


And we had the concept of a major frame, which is LCM of the periods of the tasks, periodic tasks and then the minor frames, the major frame contains an integral number of minor frames, minor cycles or frames and the tasks are assigned to the frames that was the main idea and here the periodic timer is set only during initialization do not have to set it again and again. So, let us proceed with that. One of the important design criteria here is about selecting the frame size. Now, let us see what are the constraints on the frame size that must be satisfied for the scheduler to work properly and how do we select those, let us proceed.

(Refer Slide Time: 03:06)

Three Frame Size Constraints

1. Every job needs to start and complete within a frame
 $f \geq \max(T_i), 1 \leq i \leq n$
2. Frame size f divides H (the Hyperperiod)
3. Between the release time and deadline of every job there is at least one frame





In cyclic schedulers, there are three main frame size constraints. One is that, the frame size must be larger than every job. In other words, preemption is not required, all job complete within one cycle, it is not required that a job is preempted to be part of the job to be done in another frame, that requirement is not there, so the frame is larger than every job, that is the first requirement on the frame size.

This we express as a frame size, f is greater than equal to the maximum of the execution time which we have written as T_i is the execution time, maximum of that are all tasks in the system. The second constraint on the frame size is that the major cycle should have an integral number of frames or in other words, the frame should squarely divide the major cycle and we said that the major cycle is also called as the hyperperiod.

And the third constraint is that between the release time and deadline of every job, there should be at least one frame. The first two we had discussed in some way in the last lecture and these are very intuitive, the first two with somebody can easily guess why we have these two constraints that the frame size is larger than the execution time of every task and also the frame sizes squarely divides the hyperperiod or the major cycle or the major cycle contains an integral number of frames.

Now, the third one, let us try to understand that between the release time and the deadline of every job, there must be a full frame, just draw one figure here. So, here the release time of the

task is t' . So, this is the release time or the arrival time and this is the frame tick, so just after the frame, the task has released and the deadline of the task is here, so the task is released at t' and its deadline is $t' + D_i$, where D_i is the relative deadline for that task.

And within this duration, there must be one full frame. So, $t + f$ and $t + 2f$, there is a frame here and then that is a feasible frame size. But if $t' + D_i$ were somewhere here, if $t' + D_i$ it was somewhere here, then we would, it would not be possible to execute this task. We will see the reason why that is so, in the next couple of slides, but this is an important requirement that between the release time and the deadline of the task, there must be one full frame.

(Refer Slide Time: 07:14)

Selecting a Suitable Frame Size

- **Minimum scheduling overhead and chances of inconsistency:**
 - F should be larger than each task size.
 - Sets a lower bound.
- **Minimization of table size:**
 - F should squarely divide major cycle.
 - Allows only a few discrete frame size.
- **Satisfaction of task deadline:**
 - Between the arrival of a task and its deadline, at least one full frame must exist.
 - Sets an upper bound

Handwritten notes: $M=12$, $F=12, 6, 4, 3, 2, 1$

The slide features a video inset of a man in a white shirt and glasses, and logos for IIT Bombay and NPTEL at the bottom.

So, just written down the same thing that the minimum scheduling overhead and chances of consistency, this is the requirement that the frame size should be larger than the task size. Otherwise, the scheduler becomes complicated, it has to save the context of the job and then restart the job in other frame, makes the operating system complicated and not only that, we had said that it is just one program and there is no locking mechanism etc. are not followed, because every task completes and then the next task starts, so there is no requirement for semaphores and so on.

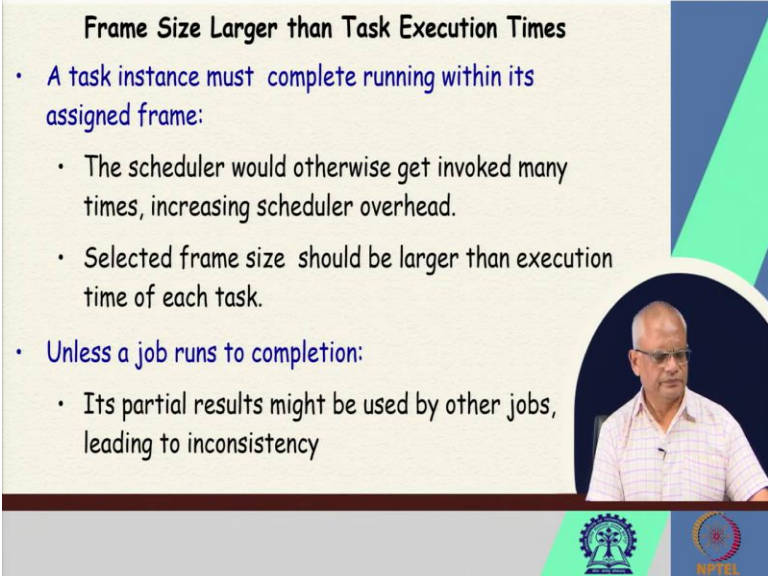
So, the frame size should be larger than each task size, is the first requirement and the frame size should be such that there are an integral number of frames in a major cycle, here since there are

integral number of frames in the major cycle, we can just store the schedule for one major cycle and keep on repeating that.

But, if there is a fraction of a frame in a major cycle, let us say 3.4 frames in a major cycle, then we cannot repeat, we need to store much larger schedule, rather than the major cycle and the second requirement where we say that the frame size should squarely divide the major cycle, so that gives us few possible frame sizes, let us say the major cycle is 12, then the integral multiples of frame can be obtained by setting the frame size, let us say the major cycle is 12, then the frame sizes can be 12 it can be 6, 4, 3, 2, 1, so the second requirement allows only a few discrete number of frame sizes.

The third one satisfaction of task deadline, there must be at least one full frame must exist and that sets an upper bound on the frame size. So, let us just discuss these three constraints in some more detail to convince you that what is this for and how do we select, let us investigate this little bit more and then we will take some examples, because this is the crux of the design of a system with a few periodic tasks to be run using a cyclic scheduler.

(Refer Slide Time: 10:53)



Frame Size Larger than Task Execution Times

- A task instance must complete running within its assigned frame:
 - The scheduler would otherwise get invoked many times, increasing scheduler overhead.
 - Selected frame size should be larger than execution time of each task.
- Unless a job runs to completion:
 - Its partial results might be used by other jobs, leading to inconsistency

The slide features a video inset of a man in a light-colored shirt speaking. At the bottom, there are logos for IIT Bombay and NPTEL.

Now, first look at frame size larger than the task execution times. If we do not let this happen that each task completes in the frame size or in other words, if the task size is greater than the frame size, then a task will not complete in a frame size. So, not only that for running one task

the scheduler runs multiple times, but also the cyclic scheduler becomes complicated and we had said that one of the motivation for using the cyclic scheduler is that it is a very simple scheduler.

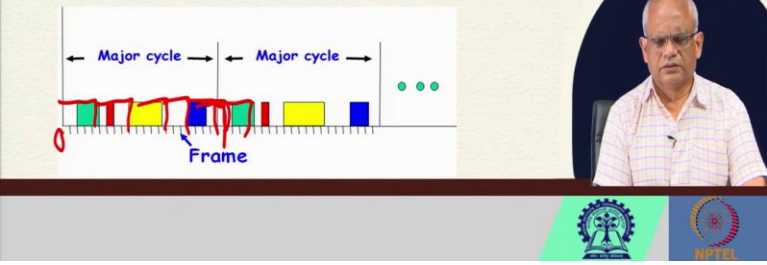
If we cannot let a task complete, we need to implement several difficult features in the operating system. For example, saving the context of a task, restarting at a later time, maintaining the consistency, because halfway it has used the resources and then got preempted and then another task ran, so it will use a partial result and that can make the final result inconsistent.

So, unless a job runs to completion, the partial results will be used by other jobs leading to inconsistency and for a job to run in to completion without any preemption, we need for the job to be completed in a single frame. So, this requirement is quite obvious, that frame size larger than task execution times to maintain the simplicity of the executing.

(Refer Slide Time: 13:04)

Frame Size Squarely Divides Major Cycle

- Unless the minor cycle squarely divides the major cycle:
 - Storing schedule for one major cycle would not be sufficient.
 - Schedules in the major cycle would not repeat.
 - This would make the size of the table large.

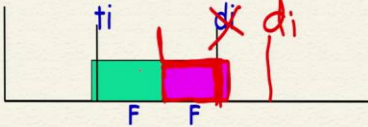


Now, the frame size squarely divides the major cycle unless it squarely divides, we cannot really store the schedule for one major cycle. Here, just see that there are an integral number of frames in the major cycle. So this is 0, the first frame, the second frame, third frame, fourth frame, fifth frame and then it starts from the zeroth frame. Now, let us imagine that there was a frame which was only partly here, then we cannot really store the schedule up to the last frame, because it would not repeat.

(Refer Slide Time: 14:15)

Between Task Arrival and its Deadline Full Frame Must Exist

- **Between the arrival of a task and its deadline:**
 - At least one full frame must exist.
- **If there is not even a single frame:**
 - The task might miss its deadline,
 - The task would not get a full frame to execute.



Now, let us look at the third constraint that between the task arrival and its deadline a full frame must exist. Because if there is not even a single frame, the task might miss its deadline, because the task will not get a full frame to execute. Let us, look at this situation given in this diagram, these are the frames, the blue, pink etc., these are different frames.

Now, let us say a certain periodic task was released at just after the frame started, since a task can be scheduled only at the beginning of a frame, it cannot be scheduled by the green one, it can at most be scheduled in the orange, sorry the pink one, so it can start only here and the deadline is before the frame, so it is not getting a full frame to complete.

So, if the task takes more time, it is a large task it cannot complete here in this frame. So, if the deadline is somewhere here, then the task gets full frame, otherwise it is only getting a part of the frame, it may not complete, now it is getting a full frame and since we know that the task execution time is smaller than the frame size, therefore it will complete if this is the situation.

(Refer Slide Time: 16:02)

Between Task Arrival and its Deadline Full Frame Must Exist

- The worst case for a task occurs when the task arrives just after a frame has started.

The top diagram shows a horizontal timeline with two frames, each of duration F , represented by green and purple blocks. A task arrival time t_i is marked with a red arrow pointing to the start of the second frame. A deadline d_i is marked with a red arrow pointing to the end of the second frame. A red bracket spans from t_i to d_i , and the text "ti misses deadline" is written to the right. The bottom diagram shows a similar timeline where the task arrival time t_i is marked with a red arrow pointing to the start of the first frame, and the deadline d_i is marked with a red arrow pointing to the end of the second frame. The text "ti deadline met" is written to the right.

8

Between Task Arrival and its Deadline Full Frame Must Exist

- The worst case for a task occurs when the task arrives just after a frame has started.

The top diagram shows a horizontal timeline with two frames, each of duration F , represented by green and purple blocks. A task arrival time t_i is marked with a red arrow pointing to the start of the second frame. A deadline d_i is marked with a red arrow pointing to the end of the second frame. A red bracket spans from t_i to d_i , and the text "ti misses deadline" is written to the right. A red handwritten equation $2F + \Delta > d_i$ is written below the diagram. The bottom diagram shows a similar timeline where the task arrival time t_i is marked with a red arrow pointing to the start of the first frame, and the deadline d_i is marked with a red arrow pointing to the end of the second frame. The text "ti deadline met" is written to the right.

But then the question that arises is that given a periodic task and the frame size, how do we know that what is the minimum duration between the start of the frame and the task, the task arrival time? Because that is the worst case, if the task arrives somewhere here, then the deadline will be somewhere here, but if the task arrives here, it will have very less time to meet the deadline.

If the tasks arrives just before the frame starts, then it can be scheduled on the frame and then that is a happy situation, the deadline can be meet here, but just see here in this situation the deadline is inside this frame and then it can start only on that frame and then it becomes difficult to meet the deadline of the task.

So, how do we express this constraint on the frame size that there must be a full frame between the start of the task and the deadline? One way to say that is if we know that this is the duration, worst case duration that is the smallest separation between the frame start and the task arrival, then we can say that $2F + \Delta > d_i$, assuming that d_i is here. So, we will just derive this expression and that will give us a constraint.

(Refer Slide Time: 18:32)

Between Task Arrival and its Deadline Full Frame Must Exist

- The minimum separation of an arrival time for t_i from a frame start:

$GCD(F, p_i)$

- Thus, for all T_i

$2F - GCD(F, p_i) \leq d_i$ must be satisfied

Now, the minimum separation between the task arrival and the frame that is the worst case situation is given by $GCD(F, p_i)$ very simple derivation, but will request you to go through the book to see what is the reason why we come with this expression $GCD(F, p_i)$.

But when we apply, we do not need those steps, how do we derive this that result is $GCD(F, p_i)$, just need to apply we should know this result, the result is $GCD(F, p_i)$ that gives us the minimum separation between the arrival time of a task and the start of the frame and then we should have $2F - GCD(F, p_i) \leq d_i$, so this the deadline should occur only after the frame.

So, that means the deadline should be larger than $2F - GCD(F, p_i)$. So, this is the important result for the third constraint slightly more involved than the other two constraints, but we need to remember this that the constraint that there must be a full frame between the arrival of a task and its deadline, can be expressed mathematically as $2F - GCD(F, p_i) < d_i$ and this is the worst case, separation the minimum separation between the arrival of a task and the frame that must be less than d_i or $d_i > 2F - GCD(F, p_i)$.

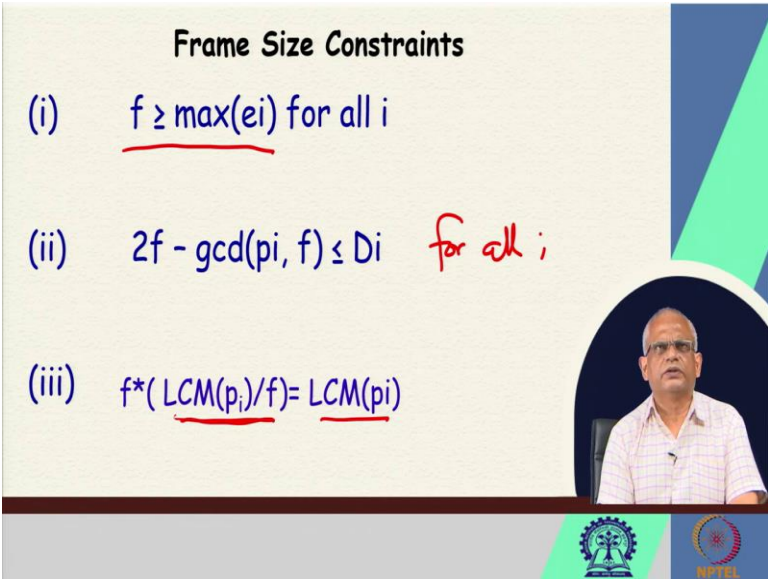
(Refer Slide Time: 20:55)

Frame Size Constraints

(i) $f \geq \max(e_i)$ for all i

(ii) $2f - \gcd(p_i, f) \leq D_i$ for all i

(iii) $f * \frac{\text{LCM}(p_i)}{f} = \text{LCM}(p_i)$



So, we can just write those three constants down mathematically, because when we are given a task scenario, we must know these expressions, so that we can easily compute the suitable frame size. The first one, the frame size is larger than the execution type of, execution time of all tasks. The second one is there must be a full frame between the arrival of the task and the deadline, we expressed that $2f - \gcd(p_i, f) \leq D_i \forall$ tasks.

And the third one we expressed that f must be an integral, sorry the LCM or the hyperperiod or the major cycle should be an integral multiple of the frame, so that we express as $f * (\text{LCM}(p_i)/f) = \text{LCM}(p_i)$ something like a C division if it is integral then we will get back LCM, if it is a fraction, then we will not get this back. So, this is the three mathematical expressions that we will use to determine a suitable frame size.

(Refer Slide Time: 22:28)

• Job=(computation time, period, relative deadline)

• T1 = (1, 6, 6)

• T2 = (2, 8, 8)

Example 1

$LCM(6,8) = 24$

• Major cycle = 24 ✓

• Frame sizes: 2, 3, 4, 6, 12, 24

• For F=2

• $2 * F - \gcd(f, P1) = 2 * 2 - 2 = 2 < 6$ Acceptable ✓

• $2 * F - \gcd(f, P2) = 2 * 2 - 2 = 2 < 8$ Acceptable ✓

The slide includes a yellow box labeled 'Example 1', a handwritten red equation $LCM(6,8) = 24$, and red checkmarks next to the major cycle and feasibility calculations. The frame sizes list has '2' boxed in red. Logos for IIT Bombay and NPTEL are visible at the bottom.

Now, let us take the first example. We have two tasks, the computation time is 1 for T1 and computation time is 2 for T2 and for the first task the period and the relative deadline are the same 6 and for T2 it is 8. Now, the first thing we need to do is to determine the major cycle and major cycle finding is straightforward it is just the LCM of the periods of the two tasks, LCM 6 comma 8 is 24, so that is the first thing we need to do is determine the major cycle.

And then we find by the second constraint, what are the feasible frame sizes? 24, 12, 6, 4, 3, 2 and possibly 1, 1 cannot be a frame size, because the frame size must be larger than the execution time of both the tasks and here the largest is 2 and therefore 2, 3, 4, 6, 12 and 24. So, these are the feasible frame sizes.

Now, we need to use the third constraint that there must be a integral, there must be a full frame between the arrival of a task and its deadline. So, let us try that out, first let us see, let us take frame size 2, so if we take frame size 2, is the, is it giving us a full frame between the arrival of a task and its deadline, the smaller ones are more likely to give the full frame and the moment it does not give in some we need not check further, because we will not get a full frame between the arrival of the task and deadline for the larger ones.

So, first let us look at the smallest one 2. So, $2 * F - \gcd(f, P1)$, so $2 * 2$ is 4 and $\gcd(2, P1)$ is 2, 2 is less than 6, therefore for T1, it is okay. Now, let us look for T2, for T2 $2 * F - \gcd(f, P2)$, this is $2 * 2 - \gcd(f, P2)$ is less than 8, so again this is satisfied. So, 2 is an acceptable frame size.

(Refer Slide Time: 25:45)

Example 1

- For F=3
 - $2 * F - \gcd(F, P1) = 6 - 3 \leq 6$ Acceptable ✓
 - $2 * F - \gcd(F, P2) = 6 - 1 \leq 8$ Acceptable ✓
- For F=4
 - $2 * F - \gcd(F, P1) = 8 - 2 \leq 6$ Acceptable
 - $2 * F - \gcd(F, P2) = 8 - 4 \leq 8$ Acceptable

But not enough frames available! ✗✗

$\frac{24}{4} = \underline{\underline{6}}$
4 frames

A = (1, 6, 6)
B = (2, 8, 8)

We can choose F=3

The slide also features a video inset of a man speaking and logos for IIT Bombay and NPTEL at the bottom.

Now, what about 3? So, for 3, if we again compute $2 * F$ is 6 and $\gcd(6,3)$ is 3, so 3 is less than equal to 6 and that is acceptable and similarly, if we test for T2, here the period is 8, so $2 * F - \gcd(3,8) = 1$, so 6 minus 1 is 5 less than 8, which is the deadline, so even for frame size 3, there is a full frame for both the tasks, so 3 is also acceptable.

But what about 4? For T1 again if we substitute, it is acceptable, for T2 also it is acceptable, but even though the constraint is satisfied for 4, but there is another problem here, the problem is that we have the LCM of 6,8 is 24 and we have the frame size 4, ok so this is gives us 6 frames in a major cycle, so we will use $F=3$.

So, here are 6 frames is actually we can, since we have only two tasks, we can schedule them, if there are more tasks, you would be in problem, so but for two task I would say that enough frames are available, so I will just remove this, so we can even use $F = 4$, so we will just revisit this example in the next lecture and we will take more examples, we will see the 3 constraints and we will see whether we can set a frame size, which is appropriate for a given problem. We will discuss examples in the next cycle; next lecture and we will also revisit this example. Thank you.